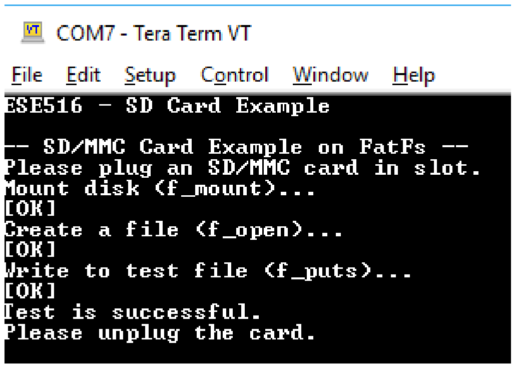
A6 Assignment – ESE516-SPRING 2019

DUE DATE: Tuesday April 2nd 2019 before 11:59pm EST (By almost midnight). To be submitted on Google Drive ON THE FOLDER OF YOUR TEAM!

Remember: Please submit your complete Altium Project on Google Teams, on a folder called A6. The project must be complete (must have everything so we can compile it!).

1.) READING [0 points, but helps with understanding]

- Review AT07175, Atmel's implementation of a bootloader on a SAM D21 processor. Source code for their implementation is here.
    1. What communication protocols can be used to update the firmware?
    2. In what conditions is the bootloader entered?
- Review AN-8185, another Atmel bootloader implementation.
- Review Atmel's CRC32 module. You'll be using this to validate data downloaded & transferred from flash.
- Review Atmel's NVM module. You'll be using this to write to the SAM D21 non volatile memory.
- Please wire the given micro SD Card module to your SAMW25 devices and use the starter code "SD_MMC_EXAMPLE_ESE516_SPRING2019" to test that it works. If it works, the system should print the following to Teraterm, and it should create a file called "sd_mmc_test.txt".



```
ESE516 - SD Card Example

-- SD/MMC Card Example on FatFs --
Please plug an SD/MMC card in slot.
Mount disk (f_mount)...
[OK]
Create a file (f_open)...
[OK]
Write to test file (f_puts)...
[OK]
Test is successful.
Please unplug the card.
```

Fig.1, output of a successful test. The SD Card should have the file "sd_mmc_test.txt"

How to wire the Micro SD Card:

Sparkfun microSD Card	**SAMW25 Port**
CS	Do not connect CS
DO	PA16
GND	GND
SCK	PA19
VCC	VCC
DI	PA18
CS	PA17

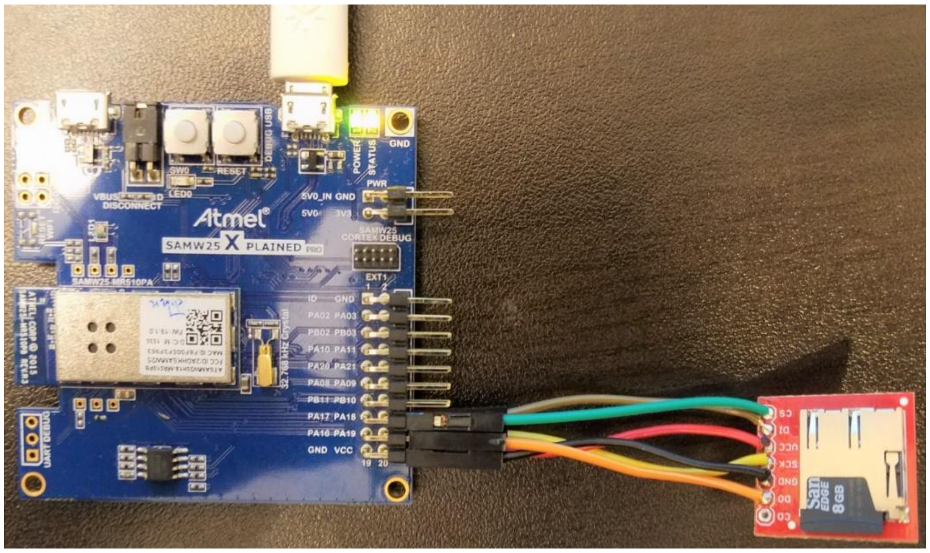


Fig.2, Example of wired SD Card module. Please note the use of SHORT WIRES.

- Review build configurations in Atmel Studio.

2.) BOOTLOADER[200 points]

Please submit a word document with the answers to these questions, and submit a zip file of your bootloader project to your group's Google Drive folder on a folder called "A6"

A.) Draw out your bootloader procedure.

a. Create a state machine / flow chart (refer to the slides & AT07175 app note)

b. How will you enter the bootloader?

i. Button press, trigger from online, scheduled every 24 hours, a combination of all of these?

c. What error cases should you handle?

i. Corrupted download or memory? Power-off mid download or during write to internal NVM or external flash? Loading a bad firmware image?

B.) Define your boot status & flash status flags as a struct to be saved on the SD Card or MCU NVM

a. What information is critical to functionality?

i. What information might be a "nice to have"?

C.) Define your partition tables for MCU Non-volatile Memory

a. Assume a bootloader of 0x2000 bytes in size, starting at address 0x0

b. You can define an area on the MCU to store flags, or do it on the SD Card. If you do it on the SD Card, please mention how you would do it (example: read if a file called bootable.txt exists and has the information needed).

D.) Create an Atmel Studio solution with two projects, Bootloader & Application Code. It should implement:

1. Jumping from the bootloader to the application code.
    i. Implement something visible to indicate the two states of the firmware -- such as LED blinking or a serial terminal printout.
    ii. Remember to review the slides & readings to learn how to implement this. It includes rebasing the stack pointer.

b. 2. Reading & writing to external flash (SD CARD) ,and verifying the content with CRC32.
    i. You can verify the CRC32 generated on the MCU with a PC program since you have access to the SD Card
    ii. Print out this information over the serial terminal.

c. 3. Reading & writing to NVM, and verifying the content with CRC32.
    i. Follow the same thoughts as the external flash in #2.
    ii. Does the SAM D21’s NVM have the same page sizes as the external flash?